LISTING OF CLAIMS

1. (Previously Presented) A software object that contains symbolic names and is executable in an execution environment for the software object, the execution environment resolving the symbolic names, the symbolic names including system symbolic names defined in the execution environment, the execution environment executing in a processor, and the software object being contained in a memory accessible to the processor, and

the software object comprising:

one or more obfuscated symbolic names that correspond to system symbolic names;

a first association between the obfuscated symbolic names and encrypted forms of the corresponding system symbolic names; and

a static watermark that has been added to the software object,

the execution environment including a second association of the encrypted forms with information needed to resolve the corresponding system symbolic names and the execution environment using the first and second associations to resolve the obfuscated symbolic names, and

using the static watermark to determine whether the software object has been altered prior to the software object being executed in the program execution environment.

2. (Previously Presented) The software object set forth in claim 1 wherein:

the static watermark's value is a digest of the software object prior to addition of the static watermark.

3. (Previously Presented) The software object set forth in claim 1 wherein the software object further comprises:

other obfuscated names that replace names defined in source code from which the software object was made.

4. (Previously Presented) The software object set forth in claim 1 wherein:

the software object is downloaded to the program execution environment for execution.

5. (Previously Presented) The software object set forth in claim 1, the software object further comprising;

an encrypted first key, the first key having been used to produce the encrypted forms of the corresponding system symbolic names,

the execution environment having access to a second key that can decrypt the first key; and the execution environment using the second key to decrypt the first key and the first key to make the encrypted forms in the second association.

6-21. (Cancelled).

22. (Previously presented) A method of protecting a software object that is executed in a host computer system from the host, the software object being executed in an execution environment for the software object in the host computer system, the execution environment loading a class that is used in executing the software object in the execution environment and the method being characterized by:

steps performed prior to execution of the software object in the execution environment comprising

replacing symbolic names in the software object that are defined in the class with obfuscated symbolic names corresponding thereto; and

making a first association between the obfuscated symbolic names and encrypted forms of the replaced symbolic names; and

steps performed during the execution of the software object in the execution environment comprising

making a second association between the encrypted forms of the symbolic names and information required to resolve the symbolic names;

adding a method to the software object that determines whether the software object has been modified by the host;

using the first and second associations to resolve the obfuscated symbolic names; and

executing the added method to determine whether the software object has been modified by the host.

23. (Previously presented) The method of protecting the software object set forth in claim 22 further characterized in that:

the steps performed prior to executing the software object further comprise the step of obfuscating other symbolic names that are not defined in the class.

24. (Previously presented) The method of protecting the software object set forth in claim 22 further characterized in that:

the method to be added is encrypted; and

the step of adding the method includes the step of decrypting the method.

25. (Previously presented) The method of protecting the software object set forth in claim 22 further characterized in that:

the software object includes information from which the method can determine whether the software object has been modified by the host; and

in the step of executing the added method, the added method uses the information to determine whether the software object has been modified by the host.

26. (Previously presented) The method of protecting the software object set forth in claim 25 further characterized in that:

the steps performed prior to executing the software object further comprise adding a static watermark to the software object; and the static watermark is the information used by the added method.

27. (Previously presented) The method of protecting the software object set forth in claim 26 further characterized in that:

in the step of adding the static watermark, the location of the static watermark in the software object is determined by a key; and

in the step of executing the added method, the added method uses the key to locate the watermark.

28. (Previously presented) The method of protecting the software object set forth in claim 22 further characterized in that:

the step of making the second association includes the steps of obtaining a key used to make the encrypted forms in the first association and using the obtained key to make the encrypted forms in the second association.

29. (Previously presented) The method set forth in claim 28 further characterized in that:

the software object includes an encrypted form of the encryption key that was used to make the encrypted forms in the first association; and

the step of obtaining the key includes the step of using a decryption key to decrypt the encrypted form of the encryption key.